Explaining EA: business architecture basics

The purpose of this document is to provide an explanation about Business Architecture (BA). Informally speaking, BA defines how work gets done within an enterprise. How work gets done is, of course, not completely unknown, but the knowledge is diffused throughout different instructions, strategic papers, reports, e-mails and in peoples' heads. The aim of BA is to make this knowledge explicit, i.e. formal, externalized and operational, so it can be used for decision making, operating control, daily work, knowledge transfer, etc.

First, it is necessary to achieve a common understanding about certain concepts (and the relationships between them) used for constructing BA. Examples of such concepts are: function, process, service, capability, etc. These concepts are used to provide different views of the enterprise. It is important that these views are coherent and that interdependencies between them are explicit.

BA is a part of Enterprise Architecture (EA), and usually BA is the least understood / developed / implemented part of EA.

Technical editor: Joanna Goodwin

Informal reviewer: Cliff Berg

1 General

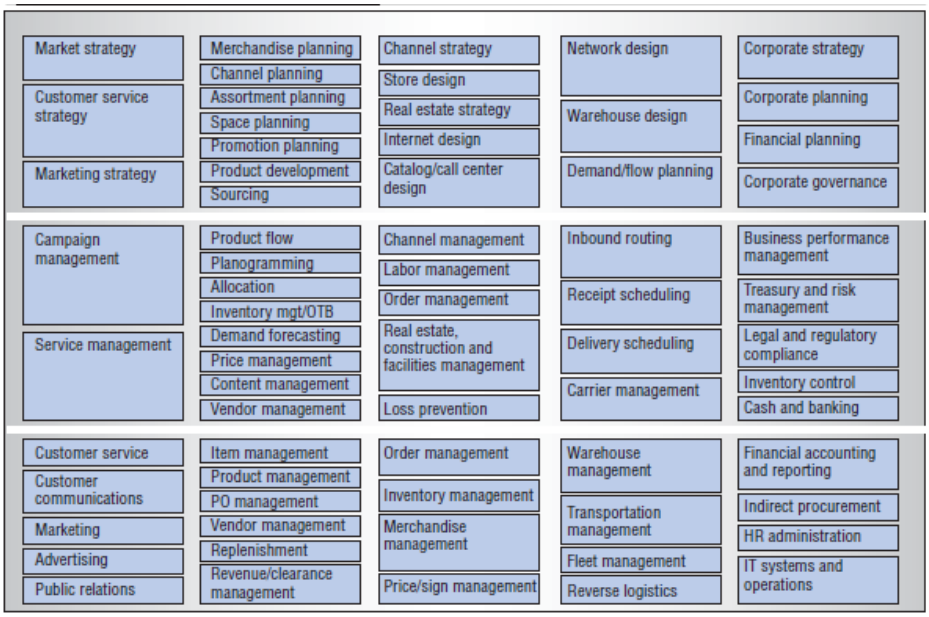
An **enterprise** creates a **result** which has **value** to a **customer** who pays for this result. The enterprise acts as a **provider** (supply-side) and the customer acts as a **consumer** (demand-side).

There is a (business) transaction between the provider and the consumer. From the point of view of the consumer (the outside-in view) the transaction is bounded by the pair "request and result", e.g. from making an order to receiving goods. From the point of view of the provider (the inside-out view) the transaction is a set of several distinct **activities** (or units of work) which function together in a logical and coordinated manner to satisfy / delight the consumer. These activities are carried out in response to the consumer's request which is an external business **event** for the provider.

2 Business functions

The collection of an enterprise's activities serves as the foundation for the discovery of business **functions** (functions deliver identifiable changes to assets). Each function is an abstract and self-contained grouping of activities that collectively satisfy a specific operational purpose (e.g. management of relationships with partners). Functions are unique within the enterprise and should not be repeated. Some functions can be decomposed into smaller groups of activities, and thus the functional view has a hierarchical structure. The structure of functions is not always the same as that of the organisation chart; in many cases, some **organisational units** can span several functions. Furthermore, organization charts may change while the function does not.

A business function typically has the suffix ‘management’ in its name (e.g. ‘Customer Relationship Management’), but it can also be a noun (e.g. ‘Marketing); usually, function name specifies something that is performed *continuously*. Some examples of business functions (from http://www-935.ibm.com/services/us/imc/pdf/g510-6163-component-business-models.pdf) are given below.



The functional view emphasizes **WHAT** the whole enterprise does to deliver value to the customer (without the organizational, application, and process constraints). Usually, the hierarchical structure of business functions is very static (with a low rate of change). Meanwhile, business processes can change more frequently as a result of business process improvement or re-engineering initiatives.

Functions and processes are different views of the enterprise. In some senses, functions are the players in a team (i.e. the enterprise), but it is not clear how they are going to play together unless there are some processes.

The functional view can be used in a number of ways:

1) to understand how organisational units are supporting each function and to identify instances where a function is supported by several organisational units (or is not supported by any organisational unit);
2) to reveal how functions are currently automated, including occurrences of where there is an overly complex use of applications (e.g. multiple applications) and when there is no automation of functions in place;
3) to understand how assets (information) flow between functions, and to map out which functions produce information, which function(s) consume information and where there is no clear understanding of information movement and ownership;
4) to clarify how business processes can be constructed;
5) to determine which business performance metrics should be used.

3 Value-streams

The collective use of activities to satisfy a customer’s request leads to the notion of a **value-stream** which is an end-to-end collection of those activities (both **value-added** and **non-value-added**) currently required by an enterprise to create a result for a customer. Value-streams are

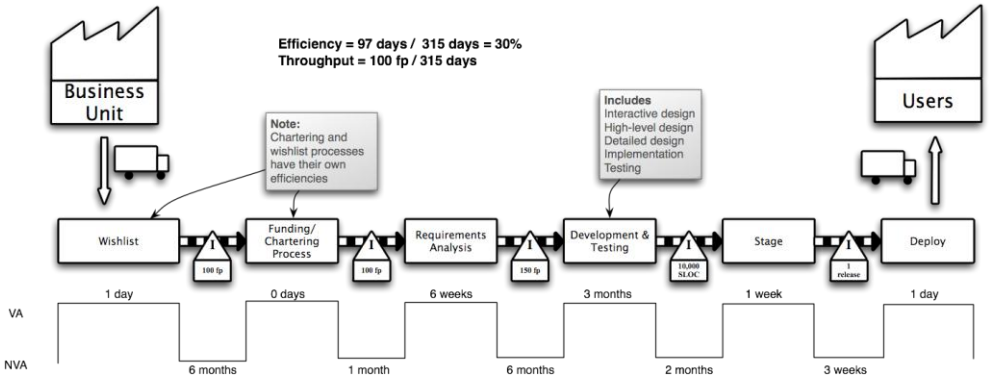
named according to an initiating event and its result. A few examples of value streams are provided below (mainly from www.enterprisebusinessarchitecture.com):

- Prospect-to-Customer
- Order-to-Cash (order fulfilment process)
- Manufacturing-to-Distribution (manufacturing process)
- Request-to-Service
- Design-to-Build
- Build-to-Order
- Build-to-Stock
- Insight-to-Strategy
- Idea-to-Concept
- Concept-to-Product
- Product-to-Launch
- Initiative-to-Results
- Relationship-to-Partnership
- Forecast-to-Plan
- Requisition-to-Payables (procurement process)
- Resource availability-to-Consumption
- Acquisition-to-Obsolescence
- Financial close-to-Reporting
- Recruitment-to-Retirement
- Awareness-to-Prevention

Value-streams are directly linked to the enterprise's aspirations – its vision and related "ends" chain (see http://www.omg.org/spec/BMM/): desired results, goals and objectives. Ideally, each value-stream should align with at least one long-range objective and its business performance metrics [**key performance indicators** (KPIs)]. For example, one objective of the success of the "Order-to-Cash" value-stream may be measured as "96% of orders delivered within 3 days". If this value-stream's actual performance is delivering only "90% of orders within 3 days" then a corrective action should be taken (e.g. a new strategic initiative is developed and its priority determined).

In addition to the reason **WHY** a value-stream exists, related to each value-stream there is an explicit **HOW** the desired results are achieved. Looking inside a value-steam reveals that there may be a few "integrated components" (or business cases – business transactions between a consumer and a provider). Usually, one of the "integrated components" is the main transaction which does the job and the others are collections of supporting/housekeeping activities. For example, "Order-to-Cash" includes "Fulfil order" (main), "Change order" (for cancellation and modification of an order by the customer), and "Review order" (for the consultation of an order by the customer).

Each "integrated component" is an ordered sequence of acts with applying functions to assets. Such a sequence is the explicit assets flow (called inputs and outputs, or I/O). KPIs and timelines associated with the sequence provide additional execution details (e.g. duration of the process from one point of I/O hand-off to another point). Thus the value-steam view provides the context (without the organisational and application constraints) for its constituent activities, e.g. what timing, level of performance, etc. are necessary to reach the objective of the complete value-stream.



An enterprise consists of a collection of value-streams. Most large enterprises can be broken down into a dozen or more value-streams. The nomenclature of value-streams differs somewhat from one enterprise to another. Within an enterprise, its value-streams are interdependent; a value-stream may rely on the results of other value-streams.

An example of this interdependency is the **value-chain** of an enterprise, i.e. a network of strategically relevant integrated components of value-streams of the enterprise. A value-chain is not a collection of independent activities but a system of interdependent activities which should be taken into account to obtain competitive advantage. This is the principal end-to-end view of the enterprise.

4 Linking WHY, WHAT and HOW

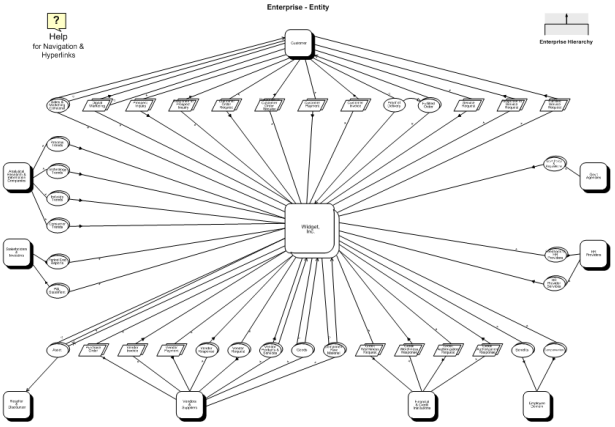
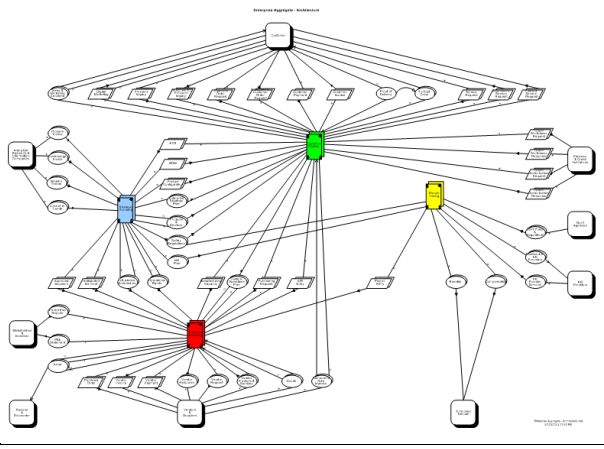
So, an enterprise's value-chain and value-streams are the high-level decomposition of the work of the (whole) enterprise into the work of many different activities, with the value flow made explicit. In such a decomposition, WHY + WHAT of the whole enterprise should be used to define WHY + WHAT of each activity. The glue between them is HOW. Let's look at a fictitious scenario.

Potential shareholders:

OK, your business model looks good. Now tell us about the operating model.

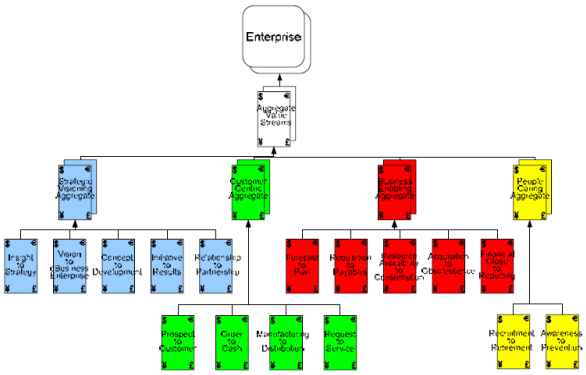
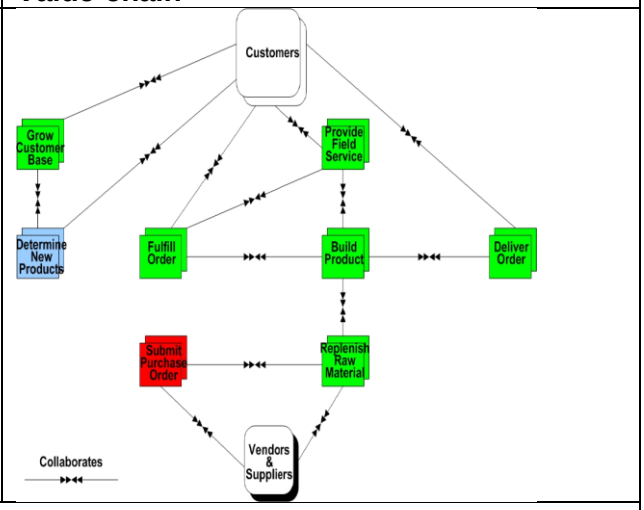
Future CEO:

Our business model is the WHY for our operating model. The latter starts by showing the relationships between the enterprise and its partners (suppliers, providers, customers, etc.) from the economic ecosystem. Within the enterprise we have identified 4 aggregations of value-streams: customer-centric (green), strategic-visioning (blue), people-caring (yellow) and business enabling (red), as well as the relationships between them.

The enterprise and related external partners	The enterprise as a set of aggregations of selected value-streams
	

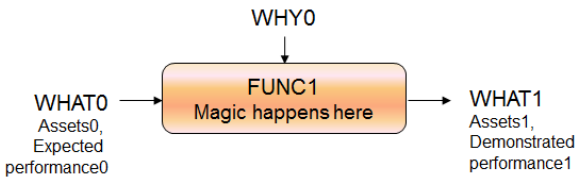
Example from www.enterprisebusinessarchitecture.com

We have developed the full nomenclature of our value-streams and their integrated components. Each value-stream is connected to a particular objective. Also, we know our value-chain (the principal end-to-end view of the enterprise composed from integrated components).

Value-streams	Value-chain
	

Example from www.enterprisebusinessarchitecture.com

So, for each value-stream (FUNC1), we know its input WHAT0 {Assets0, Expected performance0}, its output WHAT1 {Assets1, Demonstrated performance1} as well as its operating requirements WHY0.

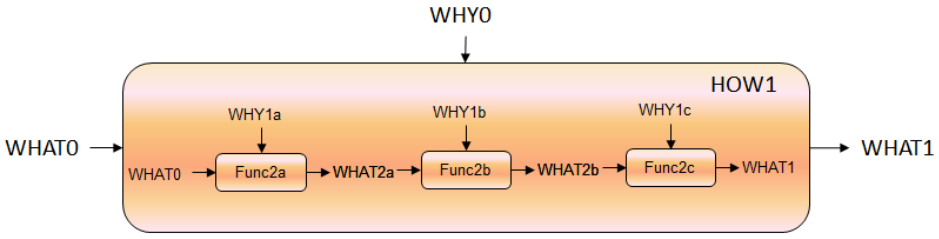


Potential shareholders:

Sounds great. And, can you assure us that FUNC1 is capable of operating as required?

Architect:

The desired performance of FUNC1 is guaranteed by its implementation (HOW1) as the explicit coordination of “smaller” functions. In some way, WHAT1 is decomposed into a set of WHAT2x. WHY0 is decomposed into a set of WHY1x, and FUNC1 is decomposed into a set of FUNC2x. They are all coordinated together. In the illustration below, the coordination is trivial, but in real cases it may be rather complex (e.g. an interaction of activities carried out by several interdependent functional roles). Also, it must be noted that the performance of the whole is not necessarily the sum of the performance of the parts.



Potential shareholders:

Please continue until all FUNC# become “manageable” activities so that they can be bought, rented, outsourced and easily implemented.

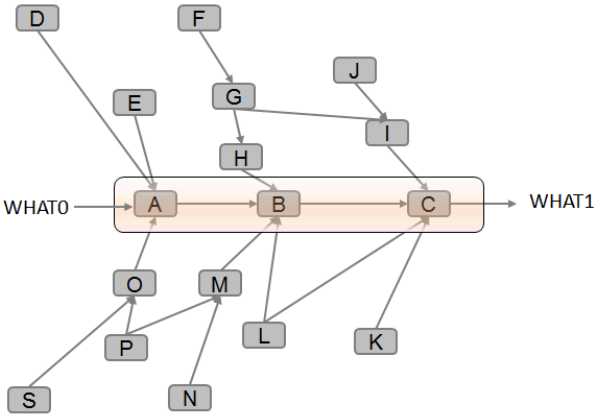
Architect:

This will involve the explicit decomposition of each value-stream to reveal the horizontal (peers) and vertical (subordinated) structure.

... Some time later ...

Architect:

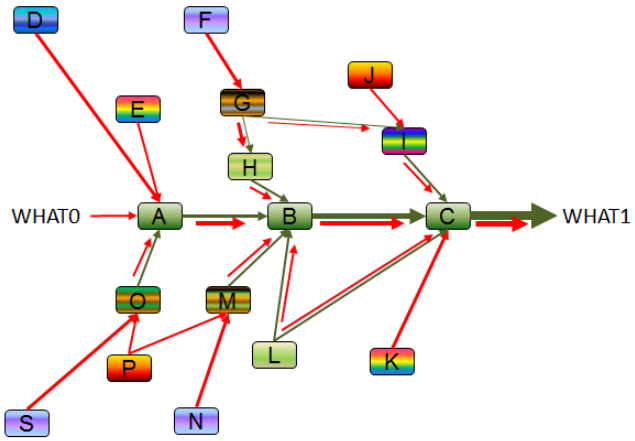
As a result of this decomposition, a directed graph can be obtained (see the figure below). This directed graph is represented as a river basin; it could also be represented as an iceberg in which the value-stream is the tip of the iceberg.



In this graph, nodes (i.e. activities) are connected by edges to show the dependencies between results (i.e. the result of activity C depends on the results of activities I, K, L and B). This means that the result of a particular activity contributes to the result of another activity (which is probably

more valuable and thus more expensive). The timing of result generation may be different: some results can be produced in advance and stored for later, some results can be produced on demand and some results can be acquired just before they are needed.

The primary importance of such a graph (called a “value & expenses basin” or “VEB”) is to represent business performance – the business wants to delight the customers (by giving them what they want to pay for) and the shareholders (by creating a profit). As shown in the figure below, different activities contribute differently to the generation of the value (green arrows) and the associated expenses (red arrows). The width of the arrows signifies the relative amount of value or expense.



Depending on the business needs, such a representation can display a particular instance of value creation or a set of instances (usually over a given period of time). In the former case, it may happen that the value disappears, e.g. a defective asset has been produced. So, the VEB is not just a “mechanical” sum of added values.

The VEB should help in the management of an enterprise. It represents a dynamic, actual and contextual contribution of different activities to the value and expenses associated with a particular result. The business can be attentive to different “tributaries” which are

a) the most value-adding,
b) the most wasteful,
c) doing worse than defined by WHY, and
d) doing better than defined by WHY.

So, how is a VEB constructed?

A VEB is not a flow of control, an event processing network (EPN), a PERT diagram or a flow of assets. It is a flow of performance metrics. *Per se*, VEB is just an externally-visible representation of internal mechanisms. Such a representation is good enough for the reactive analysis of behaviour, but is not sufficient for active control and pro-active (predictive) analytics. It is necessary to have a dynamic model which can be used for execution (e.g. simulation) and from which the VEB can be generated.

The set of “internal mechanisms” (as mentioned above) is a superposition of different coordination techniques (token-based, rule-based, event-based, data-based, etc.) which may form similar but distinct graphs relative to the VEB. Examples of those techniques are illustrated in the following.

1) An activity from one value-stream (or business process) can obtain some assets (business **objects**) which belong to another value-stream (or business process). This is pull-like

communication, e.g. the “Order-to-Cash” value-stream should know the customer’s address which is maintained by the “Prospect-to-Customer” value-stream.

2) An activity from one value-stream (or business process) can send some assets to another value-stream (or business process). The latter interprets appearing of the assets as an event to be treated. This is push-like communication. Usually, there are three ways in which this treatment can occur:
    a. a new instance should be started (e.g. for the manufacture of something) – initiating event;
    b. an existing instance, which is waiting for this event, consumes the event and continues its work (e.g. the confirmation of a payment) – solicited event;
    c. an existing instance, which does not expect this event, has to react to it – unsolicited event.

In reality, the situation is rather complicated. An enterprise may have several value-streams running in parallel. Some activities can be shared between different value-streams and some value-streams may compete for limited resources. Some activities may be outsourced or insourced, etc. All of these complexities need to be taken into account.

Furthermore, in addition to the activities, there are several other artefacts (see chapter 6) which should be defined explicitly in the model.

5 Managing the complexity of VEB

The interactions between activities reveal the different relationships between them. In order to manage the complexity, the primary interest of any architecture is to bring structure to those activities and their relationships. There are several techniques (services, capabilities, and processes) which are discussed below.

Activities which are used by a number of other activities (i.e. commonly-used functions which are the result of specialisation) are wrapped as **services** (which function as some kind of independent building blocks). A service is a consumer-facing formal representation of a self-contained provider’s repeatable set of activities which creates a result for the consumer. (It is considered that there are internal [even within an enterprise] providers and consumers.) Some functions are wrapped as services. A service may wrap several functions.

It is important that the internal functioning of a service is hidden from its consumers, so that some parts of the enterprise can be changed independently. For example, a “proper” service can be relatively easily outsourced. Services are expressed in terms of expected products, characteristics and delivery options (cost, quality, speed, capacity, geographic location, etc.) – this is the Service Level Agreement (SLA). Of course, if necessary, the internal operations of a service can be made visible, e.g. for an audit.

Complex services are created by means of the *coordination* of more simple services and/or activities (in the same way that an orchestra is a coordination of individuals and their actions). In this sense, an enterprise is a mega-service composed of a *network* of nano-services. Each service is associated with an **owner** who is responsible for delivering the promised results *in all instances* in which that service has been requested. That owner has

a) to know/estimate the demand-side *needs* (the service may have many different consumers who will be using it with different frequencies), and

b) to design/organise/create *in advance* the supply-side **capabilities** to ensure those needs are satisfied.

Capability is the *proven possession* of characteristics required to perform a particular service (to produce a particular result, which may include the required performance) and the functions which are wrapped by this service. Capability needs to "understand" the mechanics of delivering that service. The mechanics include the resources, skills, policies, powers/authorities, systems, information, other services, etc., as well as the coordination of work within the service. Capability is named after the expected result/performance (http://ingenia.wordpress.com/2010/10/19/modelling-behaviour/), e.g. "2CV".

So, how can one ensure that a service has the required characteristics? There are three control options:

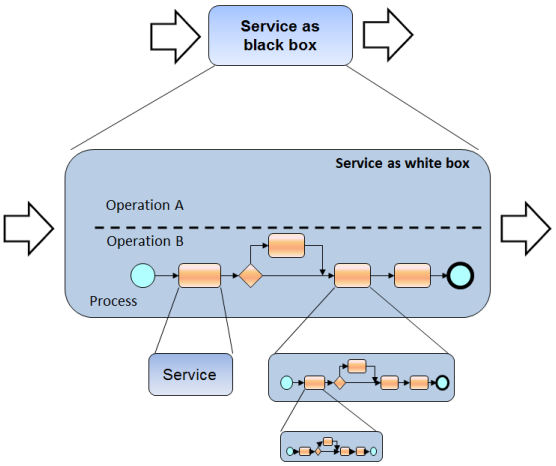
1. by contract ("re-active" approach) – acquire a service with the required characteristics, use it, check that its performance is acceptable and replace it if something is wrong with it;
2. by measurement ("active" approach) – implement a service, use it, measure it, improve or re-build it, etc.;
3. by design ("pro-active" approach) – build a service model, run a simulation test, improve the model, build the service, use it, measure it, improve it, etc.

The first option works with some **support** services, the second option can work satisfactorily with **lead** services and the third option should be used for **core business** services. The core business services can't be outsourced, can't be bought and must not be "damaged" (otherwise the enterprise may no longer function).

One of the models of the mechanics of delivering a service is a **business process** – an explicitly-defined coordination of services and/or activities to produce a particular result.

Services and processes can be considered to be intimately related since in real terms

- all processes are services,
- some operations of a service (or a very detailed function wrapped by a service) can be implemented as a (classic) process, and
- a process includes services in its implementation.



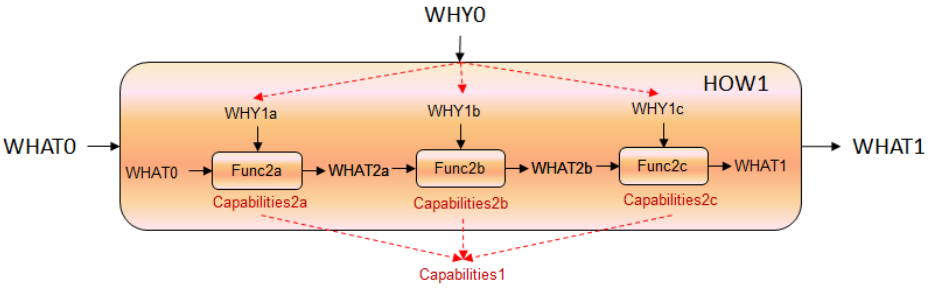
The functioning of the whole enterprise is the coordination of its activities. Such a coordination may be based on different coordination techniques (see the end of chapter 4) and be weak (e.g. a flow of data), strong (e.g. a flow of control), or intermediate (e.g. a flow of events). Clusters of activities

with strong coordination are “classic” processes (or BPMN pools). Those processes may form “business” processes if there is an intermediate coordination between them.

The explicit coordination brings several advantages.

- It allows **planning** and **simulation of** the behaviour of a service to evaluate its performance. If that service uses other services, then the demand-side needs for those services can also be evaluated.
- It can be made to be **executable**, thus guiding how work is done.
- It allows **control** that the actual behaviour of the service matches its intended behaviour, thus pro-actively detecting potential problematic situations.
- It allows the **measurement** within a service of the dynamics of different characteristics, e.g. valuing, costing, risk, etc.

So, there is a structure of services in which some services are composed from others via explicit processes. The use of explicit processes allows the objective definition of the capabilities of composed services.



6 Typology of business architecture artefacts

6.1 Motivation artefacts (why to do what)

Vision and related “ends” chain – desired result, goals, objectives

Mission and related “means” chain – course of action, strategy, tactic/projects

6.2 Value and profit proposition artefacts (what to do)

Value, value-streams, value-chain, value creation, value system, TOM?

Products or assets (tangible and intangible)

6.3 Organisation artefacts (who is doing)

Organisation structure

Governance structure

Supplier, providers, customers, and other partners

6.4 Execution artefacts (how to do what)

Process

Services

Functions

6.5 Knowledge/information artefacts (with what resources)

Terms, facts, rules, policies, etc.

6.6 Performance artefacts (how well to do what)

Capabilities

KPIs

7 Employ EA to use everything together

EA is a management tool to help the enterprise realise its vision by providing guidance and practical help for the design and evolution of the enterprise (via the enterprise models) and a coherent and proven set of principles, recommendations and practices for working with those models.

EA has three explicit parts: model, governance and management. The model part is a set of enterprise artefacts and the relationships between them. The governance part is used for the strategic improvement and self-tuning of the enterprise environment. It defines the model for a target environment and the means (a road map presented to the business as the strategy) to implement the necessary changes from the baseline model to achieve the target model. The management part supervises those changes which are carried out in different internal projects. The latter are controlled by PMO.